MobileWave: Publishing RDF Streams From SmartPhones

Yehia Abo Sedira, Riccardo Tommasini, Emanuele Della Valle

Politecnico di Milano, DEIB, Milan, Italy
{yehiamohamed.abosedera}@mail.polimi.it
{riccardo.tommasini,emanuele.dellavalle}@polimi.it

Abstract. The number of applications leveraging data from sensors on smart-phones is growing. However, they tend to focus on single Apps for *personal sensing* rather than trying to achieve a greater good (*social sensing*) by addressing challenges as App interoperability and geographical distribution of devices. In this paper, we present MobileWave a framework that allows creating, composing and publishing RDF streams from smart-phones. MobileWave extends TripleWave with a publish/subscribe middleware that collects *affluent* RDF Streams, composes them into a swollen RDF stream and publishes it on the Web.

Keywords: RDF Stream, Ontology Based Data Access, Mobile Phones

1 Introduction

Smart-phones may not have started the Web-of-Things revolution, but are now active part of it. We all became sensor when we started posting on social media, but many applications are now turning the phones into actual platform able to measure all the aspect of our life [4]. Thanks to frameworks like HealthKit[1] or GoogleFit[2] the number of applications leveraging on phones sensor capabilities is growing. However, most of them are limited to *personal sensing* [1], i.e. their scope is the phone user, e.g. location tracking and sleep monitoring. Only few apps, like Waze[3], widen their scope to pursuit *social sensing* [1], which envisions giant groups of people with a common interest exchanging data to get more value.

Can the Semantic Web actively contribute to this change of paradigm? Few years ago, the Stream Reasoning (SR) community proposed a framework, TripleWave, to publish RDF streams on the Web [5]. TripleWave works effectively when it has to transform data from a single web source, e.g. Wikipedia changes, and when it has to replay historical RDF Streams [5]. More recently, the Linked Data community proposed a way to exchange notification [2]. However, to foster *social sensing*, we need to consume data from millions of sources.

[1] https://developer.apple.com/healthkit/

[2] https://www.google.com/fit/

[3] Waze – https://www.waze.com/en-GB/ that combines GPS and social data from millions of users provide better traffic analyses.

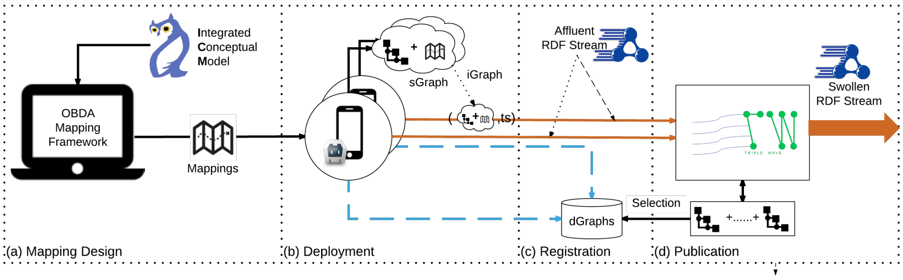

Fig. 1: MobileWave workflow: (a) R2RML mappings design; (b) mapping deployment on target; (c) affluent RDF stream registration to MobileWave middleware by *dGraph* exchange; (d) *Swollen* RDF stream publication via modified version of TripleWave.

In this paper, we propose MobileWave, a framework that allows accessing, gathering and collectively publishing RDF streams from Smart-Phone sensors. MobileWave: (i) exploits Ontology Based Data Access (OBDA) techniques on smart-phones; (ii) defines a protocol between phones and a publish-subscribe middleware; (iii) creates a composed RDF stream and publishes it through an enhanced version of TripleWave.

2 MobileWave

As in the *social sensing* vision, we aim at gathering data from millions of smart-phone users. This practically implies two problems:

1. a data integration problem, since different phones have different specifications, hardware and licensing; and
2. a scalability problem, since we cannot rely neither in high computational power nor in long battery durations.

Unfortunately, usual techniques to solve these tasks would drain the phones' battery. Phone does have enough space, memory or power to integrate data and stream directly to all the interested consumers. In the following sections we explain how MobileWave solves them using Figure 1 to visualize the workflow.

2.1 Virtual RDF Streams on the Phones

In order to solve the data integration problem (1), we ported on mobile OBDA techniques. We developed an Integrate Conceptual Model (ICM), MobileWaveOnto[4]

[4] `https://goo.gl/8ZIC2Q`

using Semantic Sensor Networks (SSN) ontology [3] and we used Ontop[5] to define R2RML[6] mappings between that ontology and the phone data. Figure 1.(a) shows the process of mapping definition using the aforementioned technologies.

In MobileWaveOnto, we modeled both static information about the device, i.e. the capabilities of the deployed sensors, and sensor observations. While the former represent quasi-static information, the latter are highly dynamic. Therefore, two kinds of mappings are possible, i.e. (a) mappings describing the device and; (b) mappings representing a sensor observation as time-varying graph. Listings 1 and 2 respectively exemplify these mappings.

Figure 1.(b) shows the deployment of the R2RML mappings on the phone. MobileWave applies the mapping by means of Apache Cordova[7], which offers a generic view over phones hardware. R2RML mappings are translated in a JSON-LD based syntax which is more convenient for Apache Cordova to combine with phone data, but it still preserves the semantic value of R2RML. Each translation includes: (1) a **Context** that maps terms to IRIs, and; (2) a **Pattern** list that contains the actual mappings from *Apache Cordova*'s JSON to RDF graphs.

2.2 From Affluents to *Swollen* RDF Streams

In order to solve the scalability problem (2), MobileWave comprises a publish/-subscribe middleware that exposes REST methods for device to register and then composes and publishes an RDF stream to be consumed. We name this stream *Swollen* RDF stream and *affluent*s RDF streams those stream that participate. To be added as an *affluent*, a device must identify itself by sharing a descriptions file. We call it description Graph (*dGraph*) and we build it by means of the (a)-type mappings described in Section 2.1 and exemplified in Listing 1. MobileWave middleware stores the *dGraph* and replies to the device with the URL where the sensor observations should be pushed.

```
"dGraph": [{
  "subject": "#GPSSensor_{{uuid}}",
  "sources": ["device", "position"],
  "properties": [{
    "predicate": "rdf:type",
    "object": "GPSSensor",
    " isLiteral ": false
  }, {
    "predicate": "ssn:onPlatform",
    "object": "SmartPhone_{{uuid}}",
    " isLiteral ": false
  }]}]
```

Listing 1: dGraph mapping example

```
"iGraph": [{
  "subject": "GPSObservation_{{time}}",
  "sources": ["device", "position"],
  "properties": [{
      "predicate": "rdf:type",
      "object": "GPSObservation",
      " isLiteral ": false
    }, {
      "predicate": "ssn:observedBy",
      "object": "GPSSensor_{{uuid}}",
      " isLiteral ": false
    }]}]
```

Listing 2: iGraph mapping example

In order to publish the *Swollen* RDF Stream we rely on two TripleWave's abstractions: the *sGraph*, a named graph that describes the stream itself and; the *iGraph*, i.e. a named graph that represents one element of the stream. We

[5] https://github.com/ontop/ontop
[6] W3C Recommendation: `https://www.w3.org/TR/r2rml/`
[7] https://cordova.apache.org/

can represent both the *affluent* RDF streams and the *Swollen* RDF using these abstractions.

MobileWave middleware is aware of all the *affluent* RDF streams descriptions (*dGraph*s) and it can select and compose them to define the *Swollen* RDF stream's *sGraph* accordingly to a given user need. For instance,it can select only the streams that come from beacons in mobile phones located in Milan.

MobileWave middleware controls the destination of the observations each *affluent* sends. Therefore, an iGraph in an affluent RDF stream directly corresponds to an iGraph in the *Swollen* RDF stream.

Figure 1.(c) and (d) show respectively the creation and publication processes. As we anticipated in Section 1 we developed a custom version of TripleWave that supports this kind of stream composition. Indeed, once the *sGraph* of the *Swollen* RDF stream is defined the publication phase, which is illustrated in Figure 1.(d), follows the same process of publishing a plain RDF Stream. From a stream consumer perspective the *Swollen* RDF stream looks like any other RDF stream on the Web.

3 Conclusion

In this paper, we presented MobileWave, a framework that enables creating, composing and collectively publishing RDF stream from mobile phones.

MobileWave uses OBDA technologies to access smart-phones sensors data and a publish/subscribe version of TripleWave that gathers data from several devices and publishes *Swollen* RDF stream on the Web.

References

1. Campbell, A.T., Eisenman, S.B., Lane, N.D., Miluzzo, E., Peterson, R.A., Lu, H., Zheng, X., Musolesi, M., Fodor, K., Ahn, G.: The rise of people-centric sensing. IEEE Internet Computing 12(4), 12–21 (2008)
2. Capadisli, S., Guy, A., Lange, C., Auer, S., Sambra, A.V., Berners-Lee, T.: Linked data notifications: A resource-centric communication protocol. In: The Semantic Web - 14th International Conference, Portorož, Slovenia, May 28 - June 1, 2017, Proceedings, Part I. pp. 537–553 (2017)
3. Compton, M., Barnaghi, P., Bermudez, L., GarcíA-Castro, R., Corcho, O., Cox, S., Graybeal, J., Hauswirth, M., Henson, C., Herzog, A., et al.: The ssn ontology of the w3c semantic sensor network incubator group. Web semantics: science, services and agents on the World Wide Web 17, 25–32 (2012)
4. Lane, N.D., Miluzzo, E., Lu, H., Peebles, D., Choudhury, T., Campbell, A.T.: A survey of mobile phone sensing. IEEE Communications Magazine 48(9), 140–150 (2010)
5. Mauri, A., Calbimonte, J.P., DellAglio, D., Balduini, M., Brambilla, M., Della Valle, E., Aberer, K.: Triplewave: Spreading rdf streams on the web. In: International Semantic Web Conference. pp. 140–149. Springer (2016)